A retrospective trust-region method for unconstrained optimization

Bastin, Fabian; Malmedy, Vincent; Toint, Philippe; Tomanos, Dimitri; Mouffe, Mélodie

Published in:
Mathematical Programming

DOI:
10.1007/s10107-008-0258-1

Publication date:
2010

Document Version
Early version, also known as pre-print

A Retrospective Trust-Region Method for Unconstrained Optimization

by F. Bastin[1], V. Malmedy[2,3], M. Mouffe[4],
Ph. L. Toint[2] and D. Tomanos[2,5]

Report 07/08 19 June 2008

[1] Computing Science and Operational Research Department
Université de Montréal
CP 6128, Succ. Centre-Ville, Montreal, Canada
Email: bastin@iro.umontreal.ca

[2] Department of Mathematics
FUNDP — University of Namur
61, Rue de Bruxelles, B-5000 Namur, Belgium
Email: vincent.malmedy@fundp.ac.be, philippe.toint@fundp.ac.be,
dimitri.tomanos@fundp.ac.be

[3] Fonds de la Recherche Scientifique – FNRS
5, Rue d'Egmont, B-1000 Bruxelles, Belgium
Research Fellow

[4] C.E.R.F.A.C.S.
42, Avenue Gustave Coriolis, F-31 057 Toulouse Cedex, France
Email: melodie.mouffe@cerfacs.fr

[5] Fonds pour la formation à la Recherche dans l'Industrie et l'Agriculture
5, Rue d'Egmont, B-1000 Bruxelles, Belgium
Research Fellow

A Retrospective Trust-Region Method for Unconstrained Optimization

F. Bastin, V. Malmedy, M. Mouffe, Ph. Toint, D. Tomanos

19 June 2008

Abstract

We introduce a new trust-region method for unconstrained optimization where the radius update is computed using the model information at the current iterate rather than at the preceding one. The update is then performed according to how well the current model retrospectively predicts the value of the objective function at last iterate. Global convergence to first- and second-order critical points is proved under classical assumptions and preliminary numerical experiments on CUTEr problems indicate that the new method is very competitive.

Keywords: unconstrained minimization, trust-region methods, convergence theory, numerical experiments.

1 Introduction

Trust-region methods are well-known techniques in nonlinear nonconvex programming, whose concept has matured over more than thirty years (for an extensive coverage, see Conn, Gould and Toint, 2000). In such methods, one considers a model m_k of the objective function which is assumed to be adequate in a "trust region", which is a neighbourhood of the current iterate x_k. This neighbourhood is often represented by a ball in some norm, whose radius Δ_k is then updated from iteration k to iteration $k+1$ by considering how well m_k predicts the objective function value at iterate x_{k+1}. In retrospect, this might seem unnatural since the new radius Δ_{k+1} will determine the region in which a possibly updated model m_{k+1} is expected to predict the value of the objective function around x_{k+1}. Our aim in this paper is to propose a variant of the trust-region algorithm that determines Δ_{k+1} according to how well m_{k+1} predicts the value of the objective function at x_k, thereby synchronizing the radius update with the change in models.

The new method is motivated by applications in adaptive techniques which exploit the information made available during the optimization process in order to vary the accuracy of the objective function computation. These techniques typically appear in the context of a noisy objective function, where noise reduction can be achieved but at a significant cost. A first trust-region method with dynamic accuracy is described in Section 10.6 of Conn et al. (2000). The main idea there is to impose a model reduction larger than some multiple of the noise evaluated at both the current and candidate iterates. A cheaper nonmonotone approach has been developed in the context of nonlinear stochastic programming by Bastin, Cirillo and Toint (2006*a*), (see also Bastin, Cirillo and Toint, 2006*b*) more specifically for the minimization of sample average approximations (Shapiro, 2003) relying on Monte-Carlo sampling, a method also known as sample-path optimization (Robinson, 1996). The main difference with respect to the work of Conn *et al.* is that it allows a reduction of the model smaller than the noise level. In both cases, the size of the model reduction is the main component to decide on the desired accuracy of the objective function: the adaptive mechanism is thus applied on the basis of past information, at the previous iterate, rather that at the current one. Our new proposal is then motivated by the hope of improving these techniques because the most relevant information on the model's quality at the current iterate would be used, instead of at the previous iterate.

This paper explores the theoretical properties and practical numerical potential of the new trust-region algorithm. We introduce the new method in Section 2, and study its convergence in the next section. Section 4 presents preliminary numerical experience on standard nonlinear problems. We conclude and examine perspectives for future research in Section 5.

2 A retrospective trust-region algorithm

We consider the unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} f(x), \tag{2.1}$$

where f is a twice-continuously differentiable objective function which maps $\mathbb{R}^n$ into $\mathbb{R}$ and is bounded below. Trust-region methods are iterative processes, which, given a starting point x_0, construct a sequence $(x_k)_{k \geqslant 0}$ of iterates hopefully converging to a solution of (2.1). At each iteration k, a twice-continuously differentiable model m_k is defined which we trust inside a (typically Euclidean) ball $\mathcal{B}_k$ of radius $\Delta_k > 0$ centred at the current iterate x_k, called the ***trust region***. A step s_k is then computed by (approximately) minimizing the model m_k inside the trust region $\mathcal{B}_k$. The trial point $x_k + s_k$ is then accepted as the next iterate x_{k+1} if and only if ρ_k, the ratio

$$\rho_k \stackrel{\text{def}}{=} \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}$$

of achieved reduction (in the objective function) to predicted reduction (in its local model m_k), is larger than a small positive constant η_1 (iteration k is then called ***successful***). In the classical framework, the trust-region radius is updated at the end of each iteration: it is left unchanged or increased if the trial point is accepted (that is if $\rho_k \geqslant \eta_1$), and decreased otherwise. In this case, the new value Δ_{k+1} is chosen in the interval $[\gamma_0 \|s_k\|, \gamma_1 \|s_k\|]$ for some constants $0 < \gamma_0 < \gamma_1 < 1$. When ρ_k is negative, a quadratic fit of the model is used (as in Conn et al., 2000, p. 783), to determine a tentative new radius whose purpose is to ensure that the next iteration is very successful in the sense that $\rho_{k+1} \geqslant \eta_2$ for some $\eta_2 \in (\eta_1, 1)$. This value is given by $\theta_k \Delta_k$, where

$$\theta_k \stackrel{\text{def}}{=} \frac{(1-\eta_2)\langle \nabla_x f(x_k), s_k \rangle}{(1-\eta_2)[f(x_k) + \langle \nabla_x f(x_k), s_k \rangle] + \eta_2 m_k(x_k + s_k) - f(x_k + s_k)}. \tag{2.2}$$

Our new algorithm differs in that the trust-region radius is updated after each successful iteration k (that is at the beginning of iteration $k+1$) on the basis of the ***retrospective*** ratio

$$\tilde{\rho}_{k+1} \stackrel{\text{def}}{=} \frac{f(x_{k+1}) - f(x_{k+1} - s_k)}{m_{k+1}(x_{k+1}) - m_{k+1}(x_{k+1} - s_k)} = \frac{f(x_k) - f(x_k + s_k)}{m_{k+1}(x_k) - m_{k+1}(x_k + s_k)}$$

of achieved to predicted changes, while continuing to use ρ_k to decide whether the trial iterate may be accepted. Our method therefore distinguishes the two roles played by ρ_k in the classical algorithm: that of deciding acceptance of the trial iterate and that of determining the radius update. It also explicitely takes into account that m_{k+1}, not m_k, is used within the trust region of radius Δ_{k+1}. Thus, when the iterate has first been accepted, that is when $\rho_k \geqslant \eta_1$, we compute this radius by either increasing the current radius or leaving it unchanged if $\tilde{\rho}_k \geqslant \tilde{\eta}_1$ or decrease it otherwise. In this last case, it is again chosen in the interval $[\gamma_0 \|s_k\|, \gamma_1 \|s_k\|]$. Moreover, when $\tilde{\rho}_k$ is negative, a quadratic fit of the model is used as above to determine a tentative new radius which will make the next iteration very successful in the sense that $\tilde{\rho}_{k+1} \geqslant \tilde{\eta}_2$ for some $\tilde{\eta}_2 \in (\tilde{\eta}_1, 1)$. This value is given by $\tilde{\theta}_{k+1}\Delta_k$, where

$$\tilde{\theta}_{k+1} \stackrel{\text{def}}{=} \frac{-(1-\tilde{\eta}_2)\langle \nabla_x f(x_{k+1}), s_k\rangle}{(1-\tilde{\eta}_2)[f(x_{k+1}) - \langle \nabla_x f(x_{k+1}), s_k\rangle] + \tilde{\eta}_2 m_{k+1}(x_k) - f(x_k)}. \tag{2.3}$$

Notice that $\tilde{\theta}_{k+1}$ uses the gradient at the new point, rather than the old one as in (2.2).

This leads to the retrospective trust-region method described as Algorithm 2.1, in which we leave the precise definitions of the model (at Step 1) and of "sufficient reduction" (at Step 3) for the next section.

Algorithm 2.1: Retrospective trust-region algorithm (RTR)

Step 0: Initialisation. An initial point x_0 and initial trust-region radius $\Delta_0 > 0$ are given. The constants η_1, $\tilde{\eta}_1$, $\tilde{\eta}_2$, γ_0, γ_1 and γ_2 are also given and satisfy $0 < \eta_1 < 1$, $0 < \tilde{\eta}_1 \leqslant \tilde{\eta}_2 < 1$ and $0 < \gamma_0 < \gamma_1 \leqslant 1 \leqslant \gamma_2$. Compute $f(x_0)$ and set $k = 0$.

Step 1: Model definition. Select a twice-continuously differentiable model m_k defined in $\mathcal{B}_k$.

Step 2: Retrospective trust-region radius update. If $k = 0$, go to Step 3. If $x_k = x_{k-1}$, then choose

$$\Delta_k = \begin{cases} \gamma_1\|s_{k-1}\| & \text{if } \rho_{k-1} \in [0, \eta_1), \\ \min[\gamma_1\|s_{k-1}\|, \max[\gamma_0, \theta_{k-1}]\Delta_{k-1}] & \text{if } \rho_{k-1} < 0, \end{cases} \tag{2.4}$$

where θ_{k-1} is defined in (2.2). Else, define

$$\tilde{\rho}_k = \frac{f(x_{k-1}) - f(x_k)}{m_k(x_{k-1}) - m_k(x_k)} \tag{2.5}$$

and choose

$$\Delta_k = \begin{cases} \max[\gamma_2\|s_{k-1}\|, \Delta_{k-1}] & \text{if } \tilde{\rho}_k \geqslant \tilde{\eta}_2, \\ \Delta_{k-1} & \text{if } \tilde{\rho}_k \in [\tilde{\eta}_1, \tilde{\eta}_2), \\ \gamma_1\|s_{k-1}\| & \text{if } \tilde{\rho}_k \in [0, \tilde{\eta}_1), \\ \min[\gamma_1\|s_{k-1}\|, \max[\gamma_0, \tilde{\theta}_k]\Delta_{k-1}] & \text{if } \tilde{\rho}_k < 0, \end{cases} \tag{2.6}$$

where $\tilde{\theta}_k$ is defined in (2.3).

Step 3: Step calculation. Compute a step s_k that "sufficiently reduces the model" m_k and such that $x_k + s_k \in \mathcal{B}_k$.

Step 4: Acceptance of the trial point. Compute $f(x_k + s_k)$ and define

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}. \tag{2.7}$$

If $\rho_k \geqslant \eta_1$, then define $x_{k+1} = x_k + s_k$ and compute $\nabla_x f(x_{k+1})$; otherwise define $x_{k+1} = x_k$. Increment k by 1 and go to Step 1.

3 Convergence theory

We now investigate the convergence properties of our algorithm. Since it can be considered as a variant of the basic trust-region method of Conn et al. (2000), we expect similar results and significant similarities in their proofs. In what follows, we have attempted to be explicit on the assumptions and properties, but to refer to Chapter 6 of this reference whenever possible.

Our assumptions are identical to those used for the basic trust-region method.

A.1 The Hessian of the objective function $\nabla_{xx} f$ is uniformly bounded, i.e. there exists a positive constant $\kappa_{\rm ufh}$ such that, for all $x \in \mathbb{R}^n$,

$$\|\nabla_{xx} f(x)\| \leqslant \kappa_{\rm ufh}.$$

A.2 The model m_k is first-order coherent with the function f at each iteration x_k, i.e. their values and gradients are equal at x_k for all k:

$$m_k(x_k) = f(x_k) \quad \text{and} \quad g_k \stackrel{\rm def}{=} \nabla_x m_k(x_k) = \nabla_x f(x_k).$$

A.3 The Hessian of the model $\nabla_{xx} m_k$ is uniformly bounded, i.e. there exists a constant $\kappa_{\rm umh} \geqslant 1$ such that, for all $x \in \mathbb{R}^n$ and for all k,

$$\|\nabla_{xx} m_k(x)\| \leqslant \kappa_{\rm umh} - 1.$$

A.4 The decrease on the model m_k is at least as much as a fraction of that obtained at the Cauchy point; i.e. there exists a constant $\kappa_{\rm mdc} \in (0,1)$ such that, for all k,

$$m_k(x_k) - m_k(x_k + s_k) \geqslant \kappa_{\rm mdc} \|g_k\| \min\left[\frac{\|g_k\|}{\beta_k}, \Delta_k\right]$$

with $\beta_k \stackrel{\rm def}{=} 1 + \max_{x \in \mathcal{B}_k} \|\nabla_{xx} m_k(x)\|$.

Note that A.4 specifies the notion of "sufficient reduction" used in Step 3 of our algorithm, while the choice of m_k in Step 1 is limited by A.2 and A.3. We also note that $s_k \neq 0$ whenever $g_k \neq 0$ because of A.4.

3.1 Convergence to First-Order Critical Points

In this section, we prove that the retrospective trust-region algorithm is globally convergent to first-order critical points, in the sense that every limit point x_* of the sequence of iterates (x_k) produced by the algorithm 2.1 satisfies

$$\nabla_x f(x_*) = 0$$

irrespective of the choice of the starting point x_0 and initial trust-region radius Δ_0.

We first give a bound on the error between the true objective function f and its current model m_k at the previous iterate x_{k-1}.

Theorem 3.1 Suppose that A.1–A.3 hold. Then we have that

$$|f(x_k) - m_{k-1}(x_k)| \leqslant \kappa_{\text{ubh}} \Delta_{k-1}^2 \tag{3.1}$$

and, if iteration $k-1$ is successful, that

$$|f(x_{k-1}) - m_k(x_{k-1})| \leqslant \kappa_{\text{ubh}} \Delta_{k-1}^2 \tag{3.2}$$

where

$$\kappa_{\text{ubh}} \stackrel{\text{def}}{=} \max[\kappa_{\text{ufh}}, \kappa_{\text{umh}}]. \tag{3.3}$$

Proof. The bound (3.1) directly results from Theorem 6.4.1 in Conn et al. (2000). We thus only prove (3.2). Because the objective function and the model are C^2 functions, we may apply the mean value theorem on the objective function f and on the model m_k, and obtain from $x_{k-1} = x_k - s_{k-1}$ that

$$\begin{aligned} f(x_{k-1}) &= f(x_k) - \langle s_{k-1}, \nabla_x f(x_k) \rangle + \tfrac{1}{2} \langle s_{k-1}, \nabla_{xx} f(\xi_k) s_{k-1} \rangle & (3.4) \\ m_k(x_{k-1}) &= m_k(x_k) - \langle s_{k-1}, g_k \rangle + \tfrac{1}{2} \langle s_{k-1}, \nabla_{xx} m_k(\zeta_k) s_{k-1} \rangle & (3.5) \end{aligned}$$

for some ξ_k, ζ_k in the segment $[x_{k-1}, x_k]$.

Because of A.2, the objective function f and the model m_k have the same value and gradient at x_k. Thus, subtracting (3.5) from (3.4) and taking absolute values yields that

$$\begin{aligned} |f(x_{k-1}) - m_k(x_{k-1})| &= \tfrac{1}{2} \left| \langle s_{k-1}, \nabla_{xx} f(\xi_k) s_{k-1} \rangle - \langle s_{k-1}, \nabla_{xx} m_k(\zeta_k) s_{k-1} \rangle \right| \\ &\leqslant \tfrac{1}{2} \left[|\langle s_{k-1}, \nabla_{xx} f(\xi_k) s_{k-1} \rangle| + |\langle s_{k-1}, \nabla_{xx} m_k(\zeta_k) s_{k-1} \rangle| \right] \\ &\leqslant \tfrac{1}{2} (\kappa_{\text{ufh}} + \kappa_{\text{umh}} - 1) \|s_{k-1}\|^2 \\ &\leqslant \tfrac{1}{2} (\kappa_{\text{ufh}} + \kappa_{\text{umh}} - 1) \Delta_{k-1}^2, & (3.6) \end{aligned}$$

where we successively used the triangle inequality, the Cauchy-Schwarz inequality, the induced matrix norm properties, A.1, A.3, and the fact that $x_k \in \mathcal{B}_{k-1}$ implies that $\|s_{k-1}\| \leqslant \Delta_{k-1}$. So (3.2) clearly holds. □

Thus the analog of Theorem 6.4.1 of Conn et al. (2000) holds in our case, where we replace the forward difference $f(x_{k+1}) - m_k(x_{k+1})$ by its retrospective variant $f(x_{k-1}) - m_k(x_{k-1})$.

As our new ratio $\tilde{\rho}_k$ uses the reduction in m_k instead of the reduction in m_{k-1}, we are interested in a bound on their difference, which is provided by this next result.

Lemma 3.2 Suppose that A.1–A.3 hold. Then we have that, for every successful iteration $k-1$,

$$\left| [m_{k-1}(x_{k-1}) - m_{k-1}(x_k)] - [m_k(x_{k-1}) - m_k(x_k)] \right| \leqslant 2\kappa_{\text{ubh}} \Delta_{k-1}^2. \tag{3.7}$$

Proof. Using the model differentiability, we apply the mean value theorem on the model m_{k-1}, and we obtain that

$$m_{k-1}(x_k) = m_{k-1}(x_{k-1}) + \langle s_{k-1}, g_{k-1} \rangle + \tfrac{1}{2} \langle s_{k-1}, \nabla_{xx} m_{k-1}(\psi_{k-1}) s_{k-1} \rangle \tag{3.8}$$

for some ψ_{k-1} in the segment $[x_{k-1}, x_k]$. Remember that (3.5) in the previous proof gives that

$$m_k(x_{k-1}) = m_k(x_k) - \langle s_{k-1}, g_k \rangle + \tfrac{1}{2} \langle s_{k-1}, \nabla_{xx} m_k(\zeta_k) s_{k-1} \rangle \tag{3.9}$$

for some ζ_k in the segment $[x_{k-1}, x_k]$. Substituting (3.8) and (3.9) inside the left-hand side of (3.7), and using A.3, the triangle inequality, the Cauchy-Schwarz inequality, and the induced matrix norm properties yield that

$$\begin{aligned} &|\, [m_{k-1}(x_{k-1}) - m_{k-1}(x_k)] - [m_k(x_{k-1}) - m_k(x_k)]\,| \\ &= \quad \left| - \langle s_{k-1}, g_{k-1} - g_k \rangle - \tfrac{1}{2} \left(\langle s_{k-1}, \nabla_{xx} m_{k-1}(\psi_{k-1}) s_{k-1} \rangle + \langle s_{k-1}, \nabla_{xx} m_k(\zeta_k) s_{k-1} \rangle \right) \right| \\ &\leqslant \quad \|s_{k-1}\| \cdot \|g_{k-1} - g_k\| + \kappa_{\rm umh} \|s_{k-1}\|^2. \end{aligned} \tag{3.10}$$

Now observe that, because of A.2, $\|g_{k-1} - g_k\| = \|\nabla_x f(x_{k-1}) - \nabla_x f(x_k)\|$. We then apply the mean value theorem on $\nabla_x f$ and obtain that

$$\nabla_x f(x_k) = \nabla_x f(x_{k-1}) + \int_0^1 \nabla_{xx} f(x_{k-1} + \alpha s_{k-1}) s_{k-1}\, d\alpha. \tag{3.11}$$

Thus the Cauchy-Schwarz inequality, and A.1 give that

$$\|g_{k-1} - g_k\| \leqslant \int_0^1 \|\nabla_{xx} f(x_{k-1} + \alpha s_{k-1})\| \cdot \|s_{k-1}\|\, d\alpha \leqslant \int_0^1 \kappa_{\rm ufh} \|s_{k-1}\|\, d\alpha = \kappa_{\rm ufh} \|s_{k-1}\|. \tag{3.12}$$

Substituting this bound in (3.10), we obtain that

$$|\, [m_{k-1}(x_{k-1}) - m_{k-1}(x_k)] - [m_k(x_{k-1}) - m_k(x_k)]\,| \leqslant (\kappa_{\rm ufh} + \kappa_{\rm umh}) \|s_{k-1}\|^2 = 2\kappa_{\rm ubh} \Delta_{k-1}^2$$

where we finally use (3.3), and the fact that $x_k \in \mathcal{B}_{k-1}$. □

We conclude from this result that the denominators in the expression of $\tilde{\rho}_k$ and ρ_{k-1} differ by a quantity which is of the same order as the error between the model and the objective function. Using this observation, we are now capable of showing that the iteration must be successful if the radius is sufficiently small compared to the gradient, and also that the trust-region radius has to increase in this case.

Theorem 3.3 Suppose that A.1–A.4 hold. Suppose furthermore that $g_k \neq 0$ and that

$$\Delta_{k-1} \leqslant \min \left[1 - \eta_1, \frac{(1 - \tilde{\eta}_2)}{(3 - 2\tilde{\eta}_2)} \right] \frac{\kappa_{\rm mdc}}{\kappa_{\rm ubh}} \|g_{k-1}\|. \tag{3.13}$$

Then iteration $k - 1$ is successful and

$$\Delta_k \geqslant \Delta_{k-1}. \tag{3.14}$$

Proof. We first apply Theorem 6.4.2 of Conn et al. (2000) to deduce that iteration $k - 1$ is successful and thus that $x_k = x_{k-1} + s_{k-1} \neq x_{k-1}$. Observe now that the constants $\tilde{\eta}_2$ and $\kappa_{\rm mdc}$ lie in the interval $(0, 1)$, which implies that

$$\frac{(1 - \tilde{\eta}_2)}{(3 - 2\tilde{\eta}_2)} < \frac{1}{2} < 1 \quad \text{and thus} \quad \kappa_{\rm mdc} \frac{(1 - \tilde{\eta}_2)}{(3 - 2\tilde{\eta}_2)} < 1. \tag{3.15}$$

The conditions (3.13), (3.15), and (3.3), combined with the definition of β_{k-1} in A.4 imply that

$$\Delta_{k-1} < \frac{\|g_{k-1}\|}{\kappa_{\rm ubh}} < \frac{\|g_{k-1}\|}{\beta_{k-1}}. \tag{3.16}$$

As a consequence, A.4 immediately gives that

$$m_{k-1}(x_{k-1}) - m_{k-1}(x_k) \geqslant \kappa_{\text{mdc}} \|g_{k-1}\| \min \left[\frac{\|g_{k-1}\|}{\beta_{k-1}}, \Delta_{k-1} \right] = \kappa_{\text{mdc}} \|g_{k-1}\| \Delta_{k-1}. \tag{3.17}$$

On the other hand, we may apply Lemma 3.2 and use the triangle inequality to obtain that

$$\begin{aligned} & \big|m_{k-1}(x_{k-1}) - m_{k-1}(x_k)\big| - \big|m_k(x_{k-1}) - m_k(x_k)\big| \\ \leqslant\ & \big|\big[m_{k-1}(x_{k-1}) - m_{k-1}(x_k)\big] - \big[m_k(x_{k-1}) - m_k(x_k)\big]\big| \\ \leqslant\ & 2\kappa_{\text{ubh}} \Delta_{k-1}^2 \end{aligned}$$

and therefore, with (3.17), that

$$\begin{aligned} |m_k(x_{k-1}) - m_k(x_k)| &\geqslant |m_{k-1}(x_{k-1}) - m_{k-1}(x_k)| - 2\kappa_{\text{ubh}} \Delta_{k-1}^2 \\ &\geqslant \kappa_{\text{mdc}} \|g_{k-1}\| \Delta_{k-1} - 2\kappa_{\text{ubh}} \Delta_{k-1}^2. \end{aligned} \tag{3.18}$$

Now (3.13) implies that $(3 - 2\tilde{\eta}_2)\kappa_{\text{ubh}} \Delta_{k-1} \leqslant (1 - \tilde{\eta}_2)\kappa_{\text{mdc}} \|g_{k-1}\|$ and thus that

$$(1 - \tilde{\eta}_2)(\kappa_{\text{mdc}} \|g_{k-1}\| - 2\kappa_{\text{ubh}} \Delta_{k-1}) \geqslant \kappa_{\text{ubh}} \Delta_{k-1} > 0. \tag{3.19}$$

We finally may apply Theorem 3.1 and deduce from A.2, (3.2), (3.18) and (3.19) that

$$|\tilde{\rho}_k - 1| = \left| \frac{f(x_{k-1}) - m_k(x_{k-1})}{m_k(x_{k-1}) - m_k(x_k)} \right| \leqslant \frac{\kappa_{\text{ubh}} \Delta_{k-1}}{\kappa_{\text{mdc}} \|g_{k-1}\| - 2\kappa_{\text{ubh}} \Delta_{k-1}} \leqslant 1 - \tilde{\eta}_2. \tag{3.20}$$

Therefore, $\tilde{\rho}_k \geqslant \tilde{\eta}_2$ and (2.6) then ensures that (3.14) holds. □

It is therefore guaranteed that the trust-region radius can not be decreased indefinitely if the current iterate is not near critically. This is ensured by the next theorem.

Theorem 3.4 Suppose that A.1–A.4 hold. Suppose furthermore that there exists a constant κ_{lbg} such that $\|g_k\| \geqslant \kappa_{\text{lbg}}$ for all k. Then there is a constant κ_{lbd} such that

$$\Delta_k \geqslant \kappa_{\text{lbd}} \tag{3.21}$$

for all k.

Proof. The proof is the same as for Theorem 6.4.3 in Conn et al. (2000) except that

$$\kappa_{\text{lbd}} = \min \left[1 - \eta_1, \frac{(1 - \tilde{\eta}_2)}{(3 - 2\tilde{\eta}_2)} \right] \frac{\gamma_1 \kappa_{\text{mdc}} \kappa_{\text{lbg}}}{\kappa_{\text{ubh}}}.$$

□

From here on, the proof for the basic trust region applies without change. We first deduce the global convergence of the algorithm to first-order critical points when it generates only finitely many successful iterations.

Theorem 3.5 Suppose that A.1–A.4 hold. Suppose furthermore that there are only finitely many successful iterations. Then $x_k = x_*$ for all sufficiently large k and x_* is first-order critical.

Proof. The same argument as in Theorem 6.4.4 in Conn et al. (2000) may be applied since the radius update is identical to that of the basic trust region method for unsuccessful iterations. □

Finally, the next two results ensure the global convergence of the algorithm to first-order critical points, by showing in a first step that at least one accumulation point of the iterates sequence is first-order critical.

Theorem 3.6 Suppose that A.1–A.4 hold. Then one has that

$$\liminf_{k\to\infty} \|\nabla_x f(x_k)\| = 0. \tag{3.22}$$

Proof. See Theorem 6.4.5 in Conn et al. (2000). □

As for the basic trust-region method, this can be extended to show that all limit points are first-order critical.

Theorem 3.7 Suppose that A.1–A.4 hold. Then one has that

$$\lim_{k\to\infty} \|\nabla_x f(x_k)\| = 0. \tag{3.23}$$

Proof. See Theorem 6.4.6 in Conn et al. (2000). □

3.2 Convergence to Second-Order Critical Points

We now investigate the possibility to exploit second-order information on the objective function, with the aim of ensuring convergence to second-order critical points, i.e. points x_* such that

$$\nabla_x f(x_*) = 0 \quad \text{and} \quad \nabla_{xx} f(x_*) \text{ is positive semidefinite.}$$

Of course, we need to clarify what we precisely mean by "second-order information". We therefore introduce the following additional assumptions:

A.5 The model is asymptotically second-order coherent with the objective function near first-order critical points, i.e.

$$\lim_{k\to\infty} \|\nabla_{xx} f(x_k) - \nabla_{xx} m_k(x_k)\| = 0 \text{ whenever } \lim_{k\to\infty} \|g_k\| = 0.$$

A.6 The Hessian of every model m_k is Lipschitz continuous, that is, there exists a constant $\kappa_{\rm lch}$ such that, for all k,

$$\|\nabla_{xx} m_k(x) - \nabla_{xx} m_k(y)\| \leqslant \kappa_{\rm lch} \|x - y\|$$

for all $x, y \in \mathcal{B}_k$.

A.7 If the smallest eigenvalue τ_k of the Hessian of the model m_k at x_k is negative, then

$$m_k(x_k) - m_k(x_k + s_k) \geqslant \kappa_{\rm sod} |\tau_k| \min(\tau_k^2, \Delta_k^2)$$

for some constant $\kappa_{\rm sod} \in (0, \frac{1}{2})$.

These assumptions are identical to those used in Sections 6.5 and 6.6 of Conn et al. (2000) for the basic trust-region method. In fact, the second-order convergence properties of the retrospective trust-region method also turn out to be exactly the same as those of the basic trust-region method, and their proofs can essentially been borrowed from this case, with the exception of Lemma 6.5.3. We therefore need to present a proof of that particular result for the new method. As we indicate below, all other results generalize without change and we only mention them for the sake of clarity.

In our analog of Lemma 6.5.3, we assume that the model reduction is eventually significant in the sense that it is at least of the same order as the error between the model and the objective function. We then show that the trust-region radius becomes asymptotically irrelevant if the steps tend to zero.

Lemma 3.8 Suppose that A.1–A.3, and A.5 hold. Suppose also that there exists a sequence (k_i) and a constant $\kappa_{\rm mqd} > 0$ such that

$$m_{k_i}(x_{k_i}) - m_{k_i}(x_{k_i} + s_{k_i}) \geqslant \kappa_{\rm mqd}\|s_{k_i}\|^2 > 0 \tag{3.24}$$

for all i sufficiently large. Finally, suppose that

$$\lim_{i\to\infty} \|s_{k_i}\| = 0.$$

Then iteration k_i is successful and

$$\tilde{\rho}_{k_i+1} \geqslant \tilde{\eta}_2 \quad \text{and} \quad \Delta_{k_i+1} \geqslant \Delta_{k_i} \tag{3.25}$$

for i sufficiently large.

Proof. We first apply Lemma 6.5.3 of Conn et al. (2000) to deduce that every iteration k_i is successful for i sufficiently large. Now, consider k_i one such iteration. The equations (3.4) and (3.5) imply that for some ξ_{k_i+1} and ζ_{k_i+1} in the segment $[x_{k_i}, x_{k_i+1}]$,

$$\begin{aligned}
|\tilde{\rho}_{k_i+1} - 1| &= \left|\frac{f(x_{k_i}) - m_{k_i+1}(x_{k_i})}{m_{k_i+1}(x_{k_i}) - m_{k_i+1}(x_{k_i+1})}\right| \\
&= \left|\frac{\langle s_{k_i}, \nabla_{xx} f(\xi_{k_i+1}) s_{k_i}\rangle - \langle s_{k_i}, \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1}) s_{k_i}\rangle}{-\langle s_{k_i}, g_{k_i+1}\rangle + \frac{1}{2}\langle s_{k_i}, \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1}) s_{k_i}\rangle}\right| \\
&\leqslant \frac{\|s_{k_i}\|^2 \cdot \|\nabla_{xx} f(\xi_{k_i+1}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\|}{\left|-\langle s_{k_i}, g_{k_i+1}\rangle + \frac{1}{2}\langle s_{k_i}, \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1}) s_{k_i}\rangle\right|}
\end{aligned} \tag{3.26}$$

where we also used the Cauchy-Schwarz inequality. By substituting $g_{k_i+1} = \nabla_x f(x_{k_i+1})$ (because of A.2) with its expression in (3.11), the denominator D of the latter fraction can be rewritten as

$$D = \left| -\left\langle s_{k_i}, g_{k_i} + \int_0^1 \nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) s_{k_i}\, d\alpha \right\rangle + \tfrac{1}{2}\langle s_{k_i}, \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1}) s_{k_i}\rangle \right|.$$

Then, replacing $-\langle s_{k_i}, g_{k_i}\rangle$ by its expression in (3.8), we obtain

$$\begin{aligned}
D = \Big| & m_{k_i}(x_{k_i}) - m_{k_i}(x_{k_i+1}) + \tfrac{1}{2}\langle s_{k_i}, \nabla_{xx} m_{k_i}(\psi_{k_i}) s_{k_i}\rangle \\
& + \tfrac{1}{2}\langle s_{k_i}, \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1}) s_{k_i}\rangle - \left\langle s_{k_i}, \int_0^1 \nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) s_{k_i}\, d\alpha \right\rangle \Big|
\end{aligned}$$

for some ψ_{k_i} in the segment $[x_{k_i}, x_{k_i+1}]$. The triangle inequality, properties of the integral, (3.24), and Cauchy-Schwarz inequality give therefore the following lower bound on D:

$$\begin{aligned}
D \;\geqslant\; & |m_{k_i}(x_{k_i}) - m_{k_i}(x_{k_i+1})| \\
& -\tfrac{1}{2}\left| \left\langle s_{k_i}, \int_0^1 [\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})] s_{k_i} \, d\alpha \right\rangle \right. \\
& \qquad \left. + \left\langle s_{k_i}, \int_0^1 [\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})] s_{k_i} \, d\alpha \right\rangle \right| \\
\geqslant\; & \kappa_{\rm mqd} \|s_{k_i}\|^2 - \tfrac{1}{2}\|s_{k_i}\| \int_0^1 \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})\| \cdot \|s_{k_i}\| \, d\alpha \\
& \qquad - \tfrac{1}{2}\|s_{k_i}\| \int_0^1 \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\| \cdot \|s_{k_i}\| \, d\alpha \\
\geqslant\; & \|s_{k_i}\|^2 (\kappa_{\rm mqd} - \tfrac{1}{2}\epsilon_i)
\end{aligned} \tag{3.27}$$

where

$$\epsilon_i \stackrel{\rm def}{=} \int_0^1 \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})\| \, d\alpha + \int_0^1 \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\| \, d\alpha.$$

The triangle inequality now implies that

$$\begin{aligned}
\|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})\| &\leqslant \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} f(x_{k_i})\| \\
&+ \|\nabla_{xx} f(x_{k_i}) - \nabla_{xx} m_{k_i}(x_{k_i})\| + \|\nabla_{xx} m_{k_i}(x_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})\|
\end{aligned} \tag{3.28}$$

and, similarly, that

$$\begin{aligned}
\|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\| &\leqslant \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} f(x_{k_i+1})\| \\
+ \|\nabla_{xx} f(x_{k_i+1}) - \nabla_{xx} m_{k_i+1}(x_{k_i+1})\| &+ \|\nabla_{xx} m_{k_i+1}(x_{k_i+1}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\|.
\end{aligned} \tag{3.29}$$

Since we now observe that

$$\begin{array}{rr}
\|(x_{k_i} + \alpha s_{k_i}) - x_{k_i}\| \leqslant \|s_{k_i}\|, & \|\psi_{k_i} - x_{k_i}\| \leqslant \|s_{k_i}\|, \\
\|(x_{k_i} + \alpha s_{k_i}) - x_{k_i+1}\| \leqslant \|s_{k_i}\|, & \|\zeta_{k_i+1} - x_{k_i+1}\| \leqslant \|s_{k_i}\|,
\end{array}$$

we may deduce that both

$$\|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})\| \;\text{ and }\; \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\|$$

converge to zero with $\|s_{k_i}\|$ because the first and third terms of the right-hand side of (3.28) and (3.29) tend to zero by continuity of the objective function's and model's Hessians, and because the middle term in the right-hand side of these inequalities also converges to zero because of A.5 and Theorem 3.7. As a consequence, $\epsilon_i \leqslant \kappa_{\rm mqd}$ when i is sufficiently large, and therefore, combining (3.26) and (3.27), and using the triangle inequality, we obtain

$$\begin{aligned}
|\tilde{\rho}_{k_i+1} - 1| &\leqslant \frac{2}{\kappa_{\rm mqd}} \|\nabla_{xx} f(\xi_{k_i+1}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\| \\
&\leqslant \frac{2}{\kappa_{\rm mqd}} \Big[\|\nabla_{xx} f(\xi_{k_i+1}) - \nabla_{xx} f(x_{k_i+1})\| \\
&\qquad + \|\nabla_{xx} f(x_{k_i+1}) - \nabla_{xx} m_{k_i+1}(x_{k_i+1})\| \\
&\qquad + \|\nabla_{xx} m_{k_i+1}(x_{k_i+1}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\| \Big]
\end{aligned} \tag{3.30}$$

By the same reasoning as for (3.28)–(3.29), the right-hand side of (3.30) tends to zero when i goes to infinity, and $\tilde{\rho}_{k_i+1}$ therefore tends to 1. It is thus larger than $\tilde{\eta}_2 < 1$ for i sufficiently large and (3.25) follows. □

As in Lemma 6.5.4 of Conn et al. (2000), we may apply this result to the entire sequence of iterates and deduce that all iterations are eventually successful and the trust-region radius bounded away from zero.

From here on, the theory in Conn et al. (2000) generalizes without significant change, yielding the following results.

Theorem 3.9 Suppose that A.1–A.5 hold and that x_{k_i} is a subsequence of the iterates generated by Algorithm RTR converging to a first-order critical point x_* where the Hessian of the objective function $\nabla_{xx} f(x_*)$ is positive definite. Suppose furthermore that $s_k \neq 0$ for all k sufficiently large. Then the complete sequence of iterates converges to x_*, all iterations are eventually very successful, and the trust-region radius Δ_k is bounded away from zero.

Proof. See Theorem 6.5.5 in Conn et al. (2000). □

We now proof that if the sequence of iterates remains in a compact set, then the existence of at least one second-order critical accumulation point is guaranteed.

Theorem 3.10 Suppose that A.1–A.7 hold and that all iterates remain in some compact set. Then there exists at least one limit point x_* of the sequence of iterates x_k produced by Algorithm RTR, which is second-order critical.

Proof. See Theorem 6.6.5 in Conn et al. (2000). □

By just strengthening the radius update rule by requiring that

$$\text{if } \tilde{\rho}_k \geqslant \tilde{\eta}_2 \text{ and } \Delta_k \leqslant \Delta_{\max}, \text{ then } \Delta_{k+1} \in [\gamma_3 \Delta_k, \gamma_4 \Delta_k] \tag{3.31}$$

for some $\gamma_4 \geqslant \gamma_3 > 1$ and some $\Delta_{\max} > 0$, we moreover obtain the second-order criticality of any limit point of the sequence of iterates generated by Algorithm RTR.

Theorem 3.11 Suppose that A.1–A.7, and (3.31) hold and let x_* be any limit point of the sequence of iterates. Then x_* is a second-order critical point.

Proof. See Theorem 6.6.8 in Conn et al. (2000). □

Thus the retrospective trust-region algorithm shares all the (interesting) convergence properties of the basic trust-region method under the same assumptions. We conclude this theory section by noting that the above convergence results are still valid if one replaces the Euclidean norm by any (possibly iteration dependent) uniformly equivalent norm, thereby allowing problem scaling and preconditioning.

4 Preliminary numerical experience

We now consider the numerical behaviour of the new algorithm, in comparison with the basic trust-region algorithm BTR (see page 116 of Conn et al. (2000)). We test both algorithms on all of the 146 unconstrained problems of the CUTEr collection (see Gould, Orban and Toint, 2003). For the problems whose dimension may be changed, we chose a reasonably small value in order not to overload the CUTEr interface with MATLAB. The starting points are the standard ones provided by the CUTEr library.

For the basic algorithm, the trust-region radius update was implemented by using the rule proposed in Conn et al. (2000), p. 783:

$$\Delta_{k+1} = \begin{cases} \max[\gamma_2\|s_k\|, \Delta_k] & \text{if } \rho_k \geqslant \eta_2, \\ \Delta_k & \text{if } \rho_k \in [\eta_1, \eta_2), \\ \gamma_1\|s_k\| & \text{if } \rho_k \in [0, \eta_1), \\ \min[\gamma_1\|s_k\|, \max[\gamma_0, \theta_k]\Delta_k] & \text{if } \rho_k < 0, \end{cases}$$

where γ_0 is fixed at 0.0625, γ_1 at 0.25, γ_2 at 2.5, η_1 at 0.05 and η_2 at 0.9 and where θ_k is given by (2.2). To avoid biasing the comparison, we have decided to make as few adaptations as possible to that rule in our retrospective variant (i.e. Step 2 in Algorithm 2.1). Thus, if iteration k is unsuccessful, i.e. $\rho_k < \eta_1$ and consequently $x_k = x_{k+1}$, we also decrease the trust-region using the above rule. If, on the contrary, iteration k is successful, i.e. $\rho_k \geqslant \eta_1$, the trust-region is updated using the procedure described in Step 2 of Algorithm 2.1 where we choose the same values as above for γ_0, γ_1 and γ_2, and take $\tilde{\eta}_1 = \eta_1 = 0.05$ and $\tilde{\eta}_2 = \eta_2 = 0.9$. The model was chosen, in both cases, to be the exact Taylor's series truncated to second-order, and the minimizer of the model inside the trust-region, was computed either exactly using the Moré-Sorensen algorithm (see Moré and Sorensen, 1983) or approximately using the Steihaug-Toint algorithm (see Steihaug, 1983, Toint, 1981). In this case, the conjugate gradient iterations are stopped if the trust-region boundary is met or as soon as the models' gradient satisfies the condition

$$\|\nabla_x m_k(x_k + s)\| \leqslant \min[0.1, \|\nabla_x m_k(x_k)\|^{1/2}]\|\nabla_x m_k(x_k)\|.$$

We considered that the iterative process converged when the Euclidean norm of the gradient became smaller than 10^{-5}. Failure was declared if the algorithm did not converge within the maximum number of 50 000 iterations.

We chose to compare the number of iterations to achieve convergence instead of the CPU time or number of function evaluations. Indeed, the cost per iteration is the same for both algorithms and they both evaluate the objective function once per iteration and compute one gradient at every successful iteration. Moreover, timings in MATLAB are often difficult to interpret.

All runs were performed in Matlab v. 7.1.0.183 (R14) Service Pack 3 on a 3.2 Ghz Intel single-core processor computer with 2 GB of RAM. Figure 4.1 represents the comparison by a performance profile of the number of iterations of the two algorithms. Performance profiles give, for every $\sigma \geqslant 1$, the proportion $p(\sigma)$ of test problems on which each considered algorithmic variant has a performance within a factor σ of the best (see Dolan and Moré, 2002, for a more complete discussion). In this figure, we have only kept the problems for which both algorithms converged to the same local solution (we excluded `BIGGS6`, `BROYDN7D`, `CHAINWOO`, `FLETCHBV`, `LOGHAIRY`, `MEYER3`, `NONCVXU2`, `NONCVXUN`, `SENSORS`, `TOINTGSS` and `VIBRBEAM`). If the subproblem is solved approximately, both algorithms failed on `PALMER1C`, `SBRYBND`, `SCOSINE`, `SCURLY10`, `SCURLY20` and `SCURLY30`. Moreover, RTR failed on `FLETCBV3`, which was solved by BTR. On the other hand, if the subproblem is solved exactly, both algorithms failed on `FLETCBV3` and BTR failed on `SCOSINE`, which was solved by RTR. Note also the number of iterations needed to reach convergence with the RTR algorithm on the highly nonconvex `HUMPS` and `LOGHAIRY` problems is much higher than for the BTR algorithm. The complete numerical results are given in Appendix A.

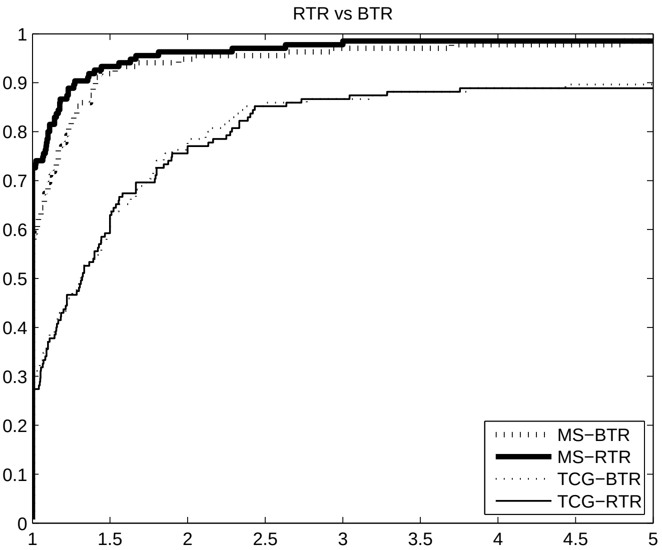

Figure 4.1: Performance profile comparing the number of iterations of the RTR and BTR algorithms

Our results show that the retrospective algorithm performs as well as the classical one and is just as reliable if the trust-region subproblem is solved approximately. However, if the problem size or structure allows an exact solution, the retrospective algorithm is then significantly more efficient (the improvement is typically of only a few iterations, but is very consistent) and just as reliable. A detailed analysis of our results shows that RTR is in general slightly more conservative than BTR in that it tends to take marginally shorter steps. However, this does not seem to alter performance in a negative way. In particular the longer steps of BTR often result in a larger proportion of unsuccesful iterations (this may be deduced from the result table since the number of unsuccessful iterations is given by the difference between the number of iterations and the number of gradient evaluations). We also note that the choice of an accurate minimization of Newton's model in the trust region also appears to be considerably more efficient than an approximate one, at least in terms of the number of iterations needed for convergence, irrespective of the choice between BTR and RTR. As a consequence, the retrospective variant is clearly at its best when the cost of evaluating the objective function and gradient dominates that of the overall iteration. Additional test not reported here also indicate that both algorithms are essentially undistinguishable when quasi-Newton approximations (SR1 or BFGS) are used instead of the true Hessian. This is perhaps not surprising since the corresponding variants, which use exact solutions of approximate models, may also be interpreted as using approximate solutions of exact models.

5 Conclusion and perspectives

We have introduced a natural variant of the basic trust-region algorithm, where the most recent model information is exploited to update the trust-region radius. We have also shown that limit points of sequences of iterates produced by the new algorithm are second-order critical points for the minimization problem. Our preliminary numerical experiments indicate that the method is advantageous when the model is good and its quality exploited by an accurate subproblem solution. Moreover this advantage is obtained at essentially zero cost.

As indicated in the introduction, this new method is especially interesting for adaptive techniques for noisy functions. The potential of the new approach is to exploit the most recent information on the noise to improve numerical performance. Research along this line is ongoing.

Other applications of the same idea are also possible across the wide class of trust-region methods, constrained and unconstrained.

References

F. Bastin, C. Cirillo, and Ph. L. Toint. An adaptive Monte-Carlo algorithm for computing mixed logit estimators. *Computational Management Science*, **3**(1), 55–80, 2006*a*.

F. Bastin, C. Cirillo, and Ph. L. Toint. Application of an adaptive Monte-Carlo algorithm to mixed logit estimation. *Transportation Research B*, **40**(7), 577–593, 2006*b*.

A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*. Number 01 *in* 'MPS-SIAM Series on Optimization'. SIAM, Philadelphia, USA, 2000.

E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, **91**(2), 201–213, 2002.

N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr, a constrained and unconstrained testing environment, revisited. *ACM Transactions on Mathematical Software*, **29**(4), 373–394, 2003.

J. J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM Journal on Scientific and Statistical Computing*, **4**(3), 553–572, 1983.

S. M. Robinson. Analysis of sample-path optimization. *Mathematics of Operations Research*, **21**(3), 513–528, 1996.

A. Shapiro. Monte Carlo sampling methods. *in* A. Shapiro and A. Ruszczyński, eds, 'Stochastic Programming', Vol. 10 of *Handbooks in Operations Research and Management Science*, pp. 353–425. Elsevier, Amsterdam, The Netherlands, 2003.

T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM Journal on Numerical Analysis*, **20**(3), 626–637, 1983.

Ph. L. Toint. Towards an efficient sparsity exploiting Newton method for minimization. *in* I. S. Duff, ed., 'Sparse Matrices and Their Uses', pp. 57–88, London, 1981. Academic Press.

Appendix A

Here is the set of results from our tests. For each problem, we report its number of variables (n), the number of iterations ($iter$), the number of gradient evaluations ($\#g$) and the best objective function value found (f). The symbol $>$ indicates that the iteration limit (fixed at 100 000) was exceeded. The columns "LS" contains a star for least-squares problems.

			Moré-Sorensen						Steihaug-Toint					
			BTR			RTR			BTR			RTR		
Name	LS	n	iter	#g	f	iter	#g	f	iter	#g	f	iter	#g	f
AKIVA		2	6	7	6.1660e+00	6	7	6.1660e+00	8	9	6.1660e+00	8	9	6.1660e+00
ALLINITU		4	7	8	5.7444e+00	7	8	5.7444e+00	5	6	5.7444e+00	5	6	5.7444e+00
ARGLINA	*	200	5	6	2.0000e+02	5	6	2.0000e+02	5	6	2.0000e+02	5	6	2.0000e+02
ARWHEAD		100	5	6	6.5947e-14	5	6	6.5947e-14	5	6	0.0000e+00	5	6	0.0000e+00
BARD	*	3	9	9	8.2149e-03	9	9	8.2149e-03	13	13	8.2149e-03	13	13	8.2149e-03
BDQRTIC	*	100	10	11	3.7877e+02	10	11	3.7877e+02	13	14	3.7877e+02	13	14	3.7877e+02
BEALE	*	2	9	9	1.9232e-16	8	8	4.5813e-14	7	8	7.3194e-12	7	8	7.3194e-12
BIGGS6	*	6	6094	4585	2.4268e-01	6021	4685	2.4268e-01	149	135	8.9467e-09	149	138	1.6487e-07
BOX3	*	3	7	8	1.5192e-11	7	8	1.5192e-11	8	9	2.3841e-15	8	9	2.3841e-15
BRKMCC		2	2	3	1.6904e-01	2	3	1.6904e-01	3	4	1.6904e-01	3	4	1.6904e-01
BROWNAL	*	200	24	20	5.3204e-23	32	27	1.2675e-15	5	6	1.4731e-09	5	6	1.4731e-09
BROWNBS	*	2	29	29	0.0000e+00	29	29	0.0000e+00	51	52	0.0000e+00	55	56	0.0000e+00
BROWNDEN	*	4	10	11	8.5822e+04	10	11	8.5822e+04	11	12	8.5822e+04	11	12	8.5822e+04
BROYDN7D		100	24	21	3.9739e+01	23	21	3.9771e+01	35	31	3.9660e+01	31	27	3.9660e+01
BRYBND	*	100	17	13	2.0687e-28	12	12	1.4121e-23	11	12	2.8661e-17	11	12	2.8661e-17
CHAINWOO	*	100	53	44	1.0000e+00	50	45	1.0000e+00	300	228	5.5035e+01	162	141	3.2191e+01
CHNROSNB	*	50	57	48	1.8917e-13	54	50	2.4837e-21	78	61	6.7337e-14	64	59	2.2256e-15
CLIFF		2	27	28	1.9979e-01	27	28	1.9979e-01	30	31	1.9979e-01	30	31	1.9979e-01
COSINE		100	6	7	-9.9000e+01	6	7	-9.9000e+01	10	10	-9.9000e+01	10	10	-9.9000e+01
CRAGGLVY		202	15	16	6.6741e+01	15	16	6.6741e+01	16	17	6.6741e+01	16	17	6.6741e+01
CUBE	*	2	37	31	9.3052e-12	35	31	1.9212e-15	44	38	1.2297e-12	42	37	1.7564e-13
CURLY10	*	50	9	10	-5.0158e+03	9	10	-5.0158e+03	18	18	-5.0158e+03	18	18	-5.0158e+03
CURLY20	*	50	8	9	-5.0158e+03	8	9	-5.0158e+03	18	18	-5.0158e+03	18	18	-5.0158e+03
CURLY30	*	50	13	13	-5.0158e+03	13	13	-5.0158e+03	17	16	-5.0158e+03	20	19	-5.0158e+03
DECONVU	*	61	25	19	1.9290e-10	19	16	1.7251e-08	22	19	3.9035e-08	22	20	3.9966e-08
DENSCHNA		2	5	6	2.2139e-12	5	6	2.2139e-12	5	6	1.2000e-15	5	6	1.2000e-15
DENSCHNB	*	2	4	5	3.3850e-16	4	5	3.3850e-16	6	7	7.9948e-14	6	7	7.9948e-14
DENSCHNC	*	2	10	11	2.1777e-20	10	11	2.1777e-20	9	10	1.8423e-13	9	10	1.8423e-13
DENSCHND	*	3	37	33	1.1392e-08	38	34	1.1392e-08	30	31	1.3753e-08	30	31	1.3753e-08
DENSCHNE	*	3	9	10	8.7102e-19	9	10	8.7102e-19	16	16	4.4587e-19	15	16	7.3809e-13
DENSCHNF	*	2	6	7	6.5132e-22	6	7	6.5132e-22	6	7	6.5132e-22	6	7	6.5132e-22
DIXMAANA		150	7	8	1.0000e+00	7	8	1.0000e+00	9	10	1.0000e+00	9	10	1.0000e+00
DIXMAANB		150	11	11	1.0000e+00	11	11	1.0000e+00	9	10	1.0000e+00	9	10	1.0000e+00
DIXMAANC		150	11	11	1.0000e+00	11	11	1.0000e+00	10	11	1.0000e+00	10	11	1.0000e+00
DIXMAAND		150	14	13	1.0000e+00	14	13	1.0000e+00	11	12	1.0000e+00	11	12	1.0000e+00
DIXMAANE		150	10	10	1.0000e+00	11	11	1.0000e+00	11	11	1.0000e+00	11	12	1.0000e+00
DIXMAANF		150	15	14	1.0000e+00	14	13	1.0000e+00	12	13	1.0000e+00	12	13	1.0000e+00
DIXMAANG		150	15	14	1.0000e+00	15	14	1.0000e+00	13	14	1.0000e+00	13	14	1.0000e+00
DIXMAANH		150	18	16	1.0000e+00	19	17	1.0000e+00	14	15	1.0000e+00	14	15	1.0000e+00
DIXMAANI		150	14	14	1.0000e+00	16	16	1.0000e+00	13	14	1.0000e+00	13	14	1.0000e+00
DIXMAANJ		150	25	21	1.0000e+00	18	16	1.0000e+00	18	17	1.0000e+00	19	18	1.0000e+00
DIXMAANK		150	23	20	1.0000e+00	19	17	1.0000e+00	22	20	1.0000e+00	20	19	1.0000e+00

			Moré-Sorensen						Steihaug-Toint					
			BTR			RTR			BTR			RTR		
Name	LS	n	iter	#g	f	iter	#g	f	iter	#g	f	iter	#g	f
DIXMAANL		150	23	20	1.0000e+00	25	22	1.0000e+00	15	16	1.0000e+00	15	16	1.0000e+00
DIXON3DQ		100	4	5	1.1710e-29	4	5	1.1710e-29	8	9	0.0000e+00	8	9	0.0000e+00
DJTL		2	105	71	-8.9515e+03	104	74	-8.9515e+03	231	161	-8.9515e+03	253	183	-8.9515e+03
DQDRTIC		100	5	6	2.3990e-28	5	6	2.3990e-28	9	10	1.7453e-17	9	10	1.7453e-17
DQRTIC		100	29	30	2.8059e-08	29	30	2.8059e-08	29	30	3.5899e-08	29	30	3.5899e-08
EDENSCH		100	19	18	6.0328e+02	20	19	6.0328e+02	17	18	6.0328e+02	17	18	6.0328e+02
EG2		100	3	4	-9.8947e+01	3	4	-9.8947e+01	3	4	-9.8947e+01	3	4	-9.8947e+01
EIGENALS	*	110	20	21	5.0766e-21	20	20	1.1113e-12	23	23	1.0531e-12	23	23	8.3333e-13
EIGENBLS	*	110	134	107	4.2412e-15	69	63	3.1853e-17	164	142	3.7937e-13	167	153	1.3427e-12
ENGVAL1		100	9	10	1.0909e+02	9	10	1.0909e+02	11	12	1.0909e+02	11	12	1.0909e+02
ENGVAL2	*	3	13	14	9.7152e-17	13	14	9.7152e-17	24	24	5.2007e-15	24	24	1.1952e-15
ERRINROS	*	50	56	48	3.9904e+01	52	47	3.9904e+01	85	79	3.9904e+01	75	72	3.9904e+01
EXPFIT	*	2	7	6	2.4051e-01	7	6	2.4051e-01	13	12	2.4051e-01	16	14	2.4051e-01
EXTROSNB	*	100	1281	1182	1.8373e-08	487	468	3.1722e-07	566	516	1.5784e-06	643	624	7.1530e-07
FLETCBV2		100	2	3	-5.1401e-01	2	3	-5.1401e-01	3	4	-5.1401e-01	3	4	-5.1401e-01
FLETCBV3		50	>	>	-3.5073e+02	>	>	-3.3920e+02	30878	30541	-1.3860e+03	>	>	-1.0286e+03
FLETCHBV		10	460	453	-2.1502e+06	1203	1151	-2.0203e+06	127	118	-2.3674e+06	257	257	-2.1109e+06
FLETCHCR		100	231	200	1.7096e-19	164	162	2.6432e-19	347	264	1.2049e-14	194	180	7.8105e-18
FMINSRF2		121	35	31	1.0000e+00	30	25	1.0000e+00	95	91	1.0000e+00	70	60	1.0000e+00
FMINSURF		121	32	27	1.0000e+00	23	19	1.0000e+00	102	98	1.0000e+00	70	59	1.0000e+00
FREUROTH	*	100	9	10	1.1965e+04	9	10	1.1965e+04	14	15	1.1965e+04	14	15	1.1965e+04
GENHUMPS	*	10	10402	9802	3.7851e-12	11624	10931	4.3255e-13	5083	4434	6.3997e-13	7075	6449	2.7198e-14
GENROSE	*	100	107	88	1.0000e+00	90	83	1.0000e+00	130	116	1.0000e+00	123	113	1.0000e+00
GENROSEB		500	460	369	1.0000e+00	327	325	1.0000e+00	585	505	1.0000e+00	498	473	1.0000e+00
GROWTHLS	*	3	96	78	1.0040e+00	79	72	1.0040e+00	183	172	1.0040e+00	171	163	1.0040e+00
GULF	*	3	30	28	1.7991e-17	32	30	3.6188e-14	40	38	3.4547e-13	44	43	3.2415e-09
HAIRY		2	64	57	2.0000e+01	116	107	2.0000e+01	96	84	2.0000e+01	91	86	2.0000e+01
HATFLDD	*	3	20	20	6.6151e-08	20	20	6.6151e-08	18	18	6.6937e-08	18	18	6.6937e-08
HATFLDE	*	3	21	21	5.1204e-07	20	20	5.1204e-07	17	17	5.1204e-07	17	17	5.1204e-07
HEART6LS	*	6	667	642	4.4113e-26	1039	1019	2.1192e-24	1528	1498	7.2910e-13	1593	1583	1.5966e-12
HEART8LS	*	8	112	95	4.6362e-17	102	88	1.7507e-13	152	143	2.0524e-20	159	154	3.8145e-14
HELIX	*	3	11	11	5.6587e-23	8	8	4.9599e-13	20	19	7.7395e-15	15	14	1.8475e-15
HIELOW		3	11	10	8.7417e+02	8	8	8.7417e+02	13	12	8.7417e+02	12	11	8.7417e+02
HILBERTA		2	3	4	2.0543e-33	3	4	2.0543e-33	3	4	1.8551e-30	3	4	1.8551e-30
HILBERTB		10	3	4	1.8835e-29	3	4	1.8835e-29	7	8	2.2225e-14	7	8	2.2225e-14
HIMMELBB		2	10	9	5.1740e-16	10	8	1.2423e-20	19	19	1.7548e-11	19	19	1.7548e-11
HIMMELBF	*	4	276	274	3.1857e+02	94	92	3.1857e+02	358	356	3.1857e+02	353	315	3.1857e+02
HIMMELBG		2	5	6	9.0327e-12	5	6	9.0327e-12	7	7	1.7308e-15	7	7	1.7308e-15
HIMMELBH		2	4	5	-1.0000e+00	4	5	-1.0000e+00	4	5	-1.0000e+00	4	5	-1.0000e+00
HUMPS	*	2	2690	2503	1.0977e-12	6856	6604	2.4027e-13	2606	2243	6.0915e-14	6265	6038	6.5371e-11
JENSMP		2	9	10	1.2436e+02	9	10	1.2436e+02	9	10	1.2436e+02	9	10	1.2436e+02
KOWOSB	*	4	11	10	3.0780e-04	11	10	3.0780e-04	12	12	3.0780e-04	12	11	3.0780e-04

			Moré-Sorensen						Steihaug-Toint					
			BTR			RTR			BTR			RTR		
Name	LS	n	iter	#g	f	iter	#g	f	iter	#g	f	iter	#g	f
LIARWHD	*	100	12	13	5.5677e−14	12	13	5.5677e−14	14	15	2.4677e−15	14	15	2.4677e−15
LOGHAIRY		2	2734	2676	1.8232e−01	9091	8167	1.8232e−01	4871	4132	5.1277e+00	7612	6953	1.8232e−01
MANCINO	*	100	14	15	1.5058e−21	16	16	4.0607e−19	20	21	1.4487e−21	20	21	1.4487e−21
MARATOSB		2	699	673	-1.0000e+00	680	667	-1.0000e+00	1882	1726	-1.0000e+00	1547	1493	-1.0000e+00
MEXHAT		2	32	30	-4.0010e−02	31	30	-4.0010e−02	19	20	-4.0010e−02	19	20	-4.0010e−02
MEYER3	*	3	481	441	8.7946e+01	416	381	8.7946e+01	686	680	8.8511e+01	693	688	8.8186e+01
MODBEALE		200	10	11	7.8240e−21	10	11	7.8240e−21	14	15	3.1114e−15	14	15	3.1114e−15
MOREBV	*	100	1	2	7.8870e−10	1	2	7.8870e−10	138	139	2.1401e−07	138	139	2.1401e−07
MSQRTALS	*	100	20	18	2.6765e−17	19	17	7.4695e−10	20	19	4.0318e−11	20	19	4.0318e−11
MSQRTBLS	*	100	16	14	1.8855e−17	16	14	9.4179e−14	21	20	4.1329e−14	21	20	4.1329e−14
NONCVXU2		100	53	47	2.3183e+02	49	41	2.3241e+02	45	40	2.3241e+02	41	34	2.3241e+02
NONCVXUN		100	42	38	2.3168e+02	41	36	2.3285e+02	44	40	2.3168e+02	41	34	2.3227e+02
NONDIA	*	100	6	7	1.4948e−18	6	7	1.4948e−18	10	11	6.5982e−15	10	11	6.5982e−15
NONDQUAR		100	15	16	2.6991e−09	15	16	2.6991e−09	110	84	2.1978e−06	97	86	1.9731e−06
OSBORNEA	*	5	37	32	5.4649e−05	30	27	5.4649e−05	64	59	5.4718e−05	82	79	5.4649e−05
OSBORNEB	*	11	21	19	4.0138e−02	21	19	4.0138e−02	22	22	4.0138e−02	22	22	4.0138e−02
OSCIPATH		8	2035	1734	1.7473e−05	2015	1804	1.4813e−05	3020	2625	3.3662e−05	2670	2488	4.3935e−05
PALMER1C		8	7	8	9.7605e−02	7	8	9.7605e−02	>	>	9.7653e−02	>	>	9.7653e−02
PALMER1D		7	7	8	6.5267e−01	7	8	6.5267e−01	23	24	6.5267e−01	23	24	6.5267e−01
PALMER2C		8	6	7	1.4369e−02	6	7	1.4369e−02	3161	3162	1.4370e−02	3161	3162	1.4370e−02
PALMER3C		8	6	7	1.9538e−02	6	7	1.9538e−02	1784	1785	1.9539e−02	1784	1785	1.9539e−02
PALMER4C		8	7	8	5.0311e−02	7	8	5.0311e−02	1538	1539	5.0312e−02	1538	1539	5.0312e−02
PALMER5C	*	6	5	6	2.1281e+00	5	6	2.1281e+00	9	10	2.1281e+00	9	10	2.1281e+00
PALMER6C	*	8	7	8	1.6387e−02	7	8	1.6387e−02	165	166	1.6388e−02	165	166	1.6388e−02
PALMER7C	*	8	9	10	6.0199e−01	9	10	6.0199e−01	6810	5734	6.0199e−01	4456	3946	6.0199e−01
PALMER8C	*	8	8	9	1.5977e−01	8	9	1.5977e−01	197	198	1.5977e−01	197	198	1.5977e−01
PENALTY1	*	100	45	44	9.0249e−04	45	44	9.0249e−04	44	41	9.0260e−04	48	44	9.0249e−04
PENALTY2	*	100	19	20	9.7096e+04	19	20	9.7096e+04	19	20	9.7096e+04	19	20	9.7096e+04
PFIT1LS	*	3	325	287	1.5734e−16	294	280	3.0857e−15	365	350	4.8505e−07	384	379	4.3509e−07
PFIT2LS	*	3	114	98	3.6218e−15	90	84	3.4229e−20	133	128	1.9620e−08	161	158	7.5351e−09
PFIT3LS	*	3	144	125	4.4639e−19	126	116	3.6432e−14	222	211	1.2519e−08	226	221	2.4788e−09
PFIT4LS	*	3	241	218	3.4144e−20	232	223	8.8142e−23	401	390	6.1391e−10	495	491	7.1420e−10
POWELLSG		4	15	16	4.6333e−09	15	16	4.6333e−09	15	16	1.2731e−08	15	16	1.2731e−08
POWER		100	24	25	1.1818e−09	24	25	1.1818e−09	25	26	1.6694e−09	25	26	1.6694e−09
QUARTC		100	29	30	2.8059e−08	29	30	2.8059e−08	29	30	3.5899e−08	29	30	3.5899e−08
ROSENBR	*	2	30	26	7.1488e−15	28	26	6.0210e−13	34	30	2.8234e−14	34	31	5.7977e−11
S308	*	2	13	12	7.7320e−01	13	12	7.7320e−01	9	10	7.7320e−01	9	10	7.7320e−01
SBRYBND	*	100	46	37	2.5620e−22	46	37	9.1262e−15	>	>	2.6525e+01	>	>	2.5463e+01
SCHMVETT		100	4	5	-2.9400e+02	4	5	-2.9400e+02	6	7	-2.9400e+02	6	7	-2.9400e+02
SCOSINE		100	>	>	-9.8840e+01	97	90	-9.9000e+01	>	>	-9.7311e+01	>	>	-9.3382e+01
SCURLY10	*	100	39	35	-1.0032e+04	46	42	-1.0032e+04	>	>	-1.0013e+04	>	>	-1.0013e+04
SCURLY20	*	100	34	30	-1.0032e+04	37	33	-1.0032e+04	>	>	-1.0032e+04	>	>	-1.0032e+04

			Moré-Sorensen						Steihaug-Toint					
			BTR			RTR			BTR			RTR		
Name	LS	n	iter	#g	f	iter	#g	f	iter	#g	f	iter	#g	f
SCURLY30	*	100	35	31	-1.0032e+04	35	31	-1.0032e+04	>	>	-1.0022e+04	>	>	-1.0021e+04
SENSORS	*	100	21	21	-1.9668e+03	24	23	-1.9668e+03	20	20	-2.0250e+03	24	22	-2.0250e+03
SINEVAL	*	2	53	46	1.9744e−25	58	52	3.3812e−36	107	93	3.6189e−18	80	73	1.4447e−21
SINQUAD		100	9	10	-4.0056e+03	9	10	-4.0056e+03	14	14	-4.0056e+03	11	12	-4.0056e+03
SISSER		2	12	13	1.0658e−08	12	13	1.0658e−08	12	13	1.2144e−08	12	13	1.2144e−08
SNAIL		2	61	61	9.3702e−13	59	60	1.2117e−14	72	72	8.6160e−17	62	63	3.6402e−18
SPARSINE		100	37	27	9.3794e−16	30	22	2.8734e−16	10	11	1.7155e−15	10	11	1.7155e−15
SPARSQUR		100	16	17	1.4795e−08	16	17	1.4795e−08	16	17	1.9872e−08	16	17	1.9872e−08
SPMSRTLS	*	100	14	13	1.2592e−13	12	11	6.1356e−12	13	13	4.6661e−14	13	13	4.6661e−14
SROSENBR	*	100	6	7	8.8993e−28	6	7	8.8993e−28	8	9	2.6078e−19	8	9	2.6078e−19
TOINTGOR		50	9	10	1.3739e+03	9	10	1.3739e+03	11	12	1.3739e+03	11	12	1.3739e+03
TOINTGSS		100	17	15	1.0102e+01	13	13	1.0204e+01	12	12	1.0102e+01	12	12	1.0102e+01
TOINTPSP		50	22	20	2.2556e+02	30	28	2.2556e+02	47	38	2.2556e+02	58	50	2.2556e+02
TQUARTIC	*	100	14	13	2.6771e−24	15	13	1.4965e−17	15	15	5.3087e−15	15	15	5.3087e−15
VARDIM		200	29	30	2.9081e−24	29	30	2.9081e−24	29	30	2.0682e−25	29	30	2.0682e−25
VAREIGVL	*	50	15	13	4.7122e−09	16	14	1.3553e−10	13	14	2.2712e−10	13	14	2.2712e−10
VIBRBEAM	*	8	49	39	1.7489e+00	51	40	1.7489e+00	668	669	1.5645e−01	960	956	1.5645e−01
WATSON	*	12	14	14	8.1544e−07	13	13	3.9067e−08	12	13	1.5973e−07	12	13	1.5973e−07
WOODS	*	4	52	44	4.6408e−15	53	47	5.1563e−17	69	59	2.0670e−13	60	54	3.8275e−17
YFITU	*	3	54	48	6.6863e−13	50	46	6.6700e−13	85	77	2.2960e−08	79	75	1.0173e−08